KEY_Lesson11_Loops2

December 10, 2021

1 Loops 2

As we saw in our last lesson, loops can be helpful in performing actions on every item in a list. But what if we don't want to iterate over *every* item in a list? Or what if we want to do something different, or something additional, to list items that meet certain requirements?

That's where conditionals, which we learned about not too long ago, become useful. In this lesson, we will learn how to combine loops with conditionals.

[1]:
```
# create a list and store it in animals
animals = ['turtle', 'penguin', 'platypus', 'zebra', 'giraffe']
```

Suppose we wanted to make a list of all the animals that aren't turtles. Within our for loop, we can write an if statement that checks if each object is not a turtle.

[2]:
```
# create a for loop that only prints the animal if it isn't a turtle
for animal in animals:
    if animal is not "turtle":
        print(animal)
```

```
penguin
platypus
zebra
giraffe
```

We can see we printed all the animals except the turtle.

What if we wanted to do something else when we came across the turtle item in our list?

[3]:
```
# edit the above for loop to print "aardvark" when the item says "turtle"
for animal in animals:
    if animal is not "turtle":
        print(animal)
    else:
        print("aardvark")
```

```
aardvark
penguin
platypus
```

```
zebra
giraffe
```

Instead of printing out the word turtle, we printed out the word aardvark!

We also can write our loops so that some code is in a conditional, and some is not. Let's say we want to print something before every entry that is not a turtle.

[4]:
```
# use an if statement inside a for statement to print "the following animal is␣
 ↪not a turtle"
# before every animal that is not a turtle
for animal in animals:
    if animal is not "turtle":
        print("The following animal is not a turtle")
    print(animal)
```

```
turtle
The following animal is not a turtle
penguin
The following animal is not a turtle
platypus
The following animal is not a turtle
zebra
The following animal is not a turtle
giraffe
```

Maybe there are some animals we don't want to print out. We learned in the previous lesson that we can create a list of the indices we want to print out. But what if we don't know what indices correspond to the items we do and don't want to print? We could use an `if/else` statement, or we could use something called `continue`.

The `continue` keyword skips to the following item in a list. Wherever you put `continue` in your for loop, any code within the loop that follows the `continue` statement won't execute for that item.

[10]:
```
# use continue to print out all animals that aren't a penguin or a turtle
for animal in animals:
    if animal is 'penguin' or animal is 'turtle':
        continue
    print(animal)
```

```
platypus
zebra
giraffe
```

Another keyword that is useful for controlling your for loop is `break`. If you want your for loop to stop running after a certain condition is true, you can use `break`. Once `break` is called, the foor loop stops—no other code runs for the current item, and the for loop won't continue to the rest of the items in the list.

[11]:
```
# use break to only print animals until we get to the zebra
for animal in animals:
    print(animal)
    if animal is 'zebra':
        break
```

```
turtle
penguin
platypus
zebra
```

We see that “giraffe” wasn’t printed. That’s because our for loop was broken after we got to the zebra.“

Great job! You just learned even more about loops. You learned: - How to use if statements within a for loop - How to use `continue` to skip certain items in a list - How to use `break` to stop running code in a for loop after a condition has been met